

**METHOD AND APPARATUS FOR GENERATING TRANSACTION-BASED
STIMULUS FOR SIMULATION OF VLSI CIRCUITS USING EVENT COVERAGE
ANALYSIS**

Clinton M. Ramsey
3316 Starlight Trail
Plano, Texas 75023
Citizenship: U.S.A.

RELATED APPLICATIONS

The present application is related to concurrently filed, co-pending, and commonly assigned U.S. Patent Application Serial No. [Attorney Docket Number 10002360-1], entitled "METHOD AND APPARATUS FOR ENCODING AND GENERATING TRANSACTION-BASED STIMULUS FOR SIMULATION OF VLSI CIRCUITS," the disclosure of which is hereby incorporated herein by reference.

TECHNICAL FIELD

This invention relates in general to genetic algorithmic directed generation of transaction stimuli states and more particularly to the construction of new test stimuli from already occurring test stimuli via a genetic algorithm, thereby reaching new constructed states that are not generated by already occurring stimuli.

BACKGROUND

Simulating integrated circuits has proved to be a very useful process for finding circuit design problems before product release. In particular, known testing techniques have been directed to CPUs (central processing units) and associated buses. Typically, in testing VLSI (very large-scale integrated) circuits such as these, a large number of different test cases are processed in order to uncover problem states. An example of test cases may be found in U.S. Patent No. 5,956,476 to Ransom et al., which is incorporated herein by reference. In previous approaches, a large number of cases might be randomly generated by specifying a weighted mix of transaction (e.g., read, write) types. For example, a mixture of 20% reads and 80% writes, might be generated. However, the timing relationship between any of the events is neither specified nor controllable when random generation is utilized.

FIGURE 1 illustrates an exemplary topology utilized by known circuit simulation techniques. VLSI 101 is associated with two separate stimuli: internal bus 102 and external bus 104 of bus interface unit 103. It is appropriate to generate a number of test states associated with these two stimuli. Most pairs of stimuli do not lead to any problems. Accordingly, it is appropriate to isolate the problem states (i.e., pairs of stimuli) that give rise to circuit malfunction. In addition, it shall be appreciated that there is an additional degree of freedom associated with problem states. Specifically, problem states that are separated in time may nevertheless lead to problematic circuit behavior.

As previously noted, test cases for application to such a system topology may be randomly generated. Random generation is illustrated in Cox, P., et al. "Statistical Modeling for Efficient Parametric Yield Estimation of MOS VLSI Circuits", IEEE Trans. Electron. Devices, vol. ED-32 No. 2, 471-478, 1985, which is incorporated herein by reference. Another example of random generation of test cases is included in U.S. Patent No. 6,018,623 to Chang et al., which is incorporated herein by reference.

Also, known techniques utilized to ascertain problem states include manual development of test cases. For example, a transaction might be set up on an external bus at time t_0 and then have a transaction injected onto the internal bus at a time $t = t_0 - t_A$. Also, t_A is initially set to equal t_1 and successively varied until t equals time t_0 . Also, t_A may be varied until it reaches $-t_1$. The hypothesis is that if there is any important timing relationship between the two transactions, that relationship would be discovered by exhaustively iterating the relative times of the two transactions.

Random generation of test states is fairly automatic. However, the amount of computational effort necessary to cover desired events can be very large. Also, random generation is problematic in that it is often difficult to repeat problem states. For example, if something changes in the execution of the model of the circuit under test, a purely random string of transactions may not re-trigger that event. Alternatively, handwritten generation of test states may identify problem states more easily. However, handwritten generation of states requires extensive hand coding by someone who has extensive knowledge of the system under test. Specifically, this technique requires significant input from highly skilled personnel.

5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203

SUMMARY OF THE INVENTION

Accordingly, there is a need in the art for the automatic generation of test states while fully simulating logic block circuits. In particular, there is a need in the art for a system and method of test state generation that requires minimal input from highly skilled personnel while providing clear and repeatable test results.

5 The present invention is directed to a system and method utilized to construct new test states from an existing set of test states and result data. The system and method are preferably implemented via genetic algorithmic generation of test states to provide a test set which, in turn, generates output states. A genetic algorithm is an algorithm that utilizes existing non-optimal solutions that provide a degree of utility to form other solutions that provide a greater degree of utility. Specifically, a genetic algorithm analyzes existing solutions and characteristics associated with those solutions. Advantageous characteristics from a number of existing solutions are combined to form new solutions which, in turn, are applied to a specific application. A genetic algorithm generally involves creating a systematic encoding nomenclature to describe solutions. Furthermore, a genetic algorithm generally involves providing a mechanism to evaluate solutions. Also, genetic algorithms provide a mechanism to describe characteristics of the various solutions.

10 In the present system and method, the genetic algorithmic generation is preferably applied to the simulation of VLSI logic circuit blocks. The system and method preferably utilize the solution nomenclature set forth in related application, U.S. Patent Application Serial No. [Attorney Docket Number 10002360-1], entitled "METHOD AND APPARATUS FOR ENCODING AND GENERATING TRANSACTION-BASED STIMULUS FOR SIMULATION OF VLSI CIRCUITS," the disclosure of which is incorporated herein by reference. Also, the system and method preferably generate a number of original test cases (solutions) as described in the preceding related application. Alternatively, original test cases may be generated randomly or by directed pairing of instructions. This aggregate of solutions is provided to a circuit simulator. The results of the simulator are preferably maintained in a matrix or table. The results preferably detail the number of times that particular logic states or events associated with the VLSI block have been stimulated by particular test cases. The aggregate of solutions and the simulation results are then analyzed by the genetic algorithm. The genetic algorithm preferably identifies states associated with the circuit simulation that have not been produced by the original test cases. The genetic

algorithm then combines characteristics of various test cases to generate new test cases. The new test cases are provided to the circuit simulator thereby providing a higher degree of confidence that the entire VLSI chip design has been simulated.

5 The foregoing has outlined rather broadly the features and technical advantages of the present invention in order that the detailed description of the invention that follows may be better understood. Additional features and advantages of the invention will be described hereinafter which form the subject of the claims of the invention. It should be appreciated by those skilled in the art that the conception and specific embodiment disclosed may be readily utilized as a basis for modifying or designing other structures for carrying out the same
10 purposes of the present invention. It should also be realized by those skilled in the art that such equivalent constructions do not depart from the spirit and scope of the invention as set forth in the appended claims. The novel features which are believed to be characteristic of the invention, both as to its organization and method of operation, together with further objects and advantages will be better understood from the following description when considered in
15 connection with the accompanying figures. It is to be expressly understood, however, that each of the figures is provided for the purpose of illustration and description only and is not intended as a definition of the limits of the present invention.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention, reference is now made to the following descriptions taken in conjunction with the accompanying drawing in which:

FIGURE 1 illustrates an emulated VLSI circuit topology;

FIGURE 2 illustrates several "strides," with each stride being associated with a transaction type at the stride interval;

FIGURE 3 illustrates a table of events versus test cases indicating the number of times that events have been stimulated by the various test cases;

FIGURE 4A illustrates test cases composed of various instructions;

FIGURE 4B illustrates a weighted table of instructions; and

FIGURE 5 depicts a block diagram of a computer system which is adapted to use the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

10 15 20 25 30

The present invention preferably utilizes a genetic algorithm. Typically, genetic algorithms search for a single best solution. For example, a genetic algorithm may be used to design an optimum jet engine by varying parameters such as the number of fan blades, size of fan blades, and/or the like. For test case generation associated with VLSI block circuit simulation, there is no single test case which fully tests the circuit under examination. Accordingly, the present invention preferably utilizes a genetic algorithm that generates a large population of diverse solutions, not a single solution. Moreover, the genetic algorithm preferably utilizes a plurality of solutions to generate new solutions. The genetic algorithm preferably utilizes a whole simulation history, which may be a large population of solutions, all of which might be “near” the desired output event. New test states are preferably generated based on the simulation history.

The present invention is preferably implemented via genetic algorithmic generation of test states to provide a test set which, in turn, generates output states. A genetic algorithm is an algorithm that utilizes existing non-optimal solutions that provide a degree of utility to form other solutions that provide a greater degree of utility. Specifically, a genetic algorithm analyzes existing solutions and characteristics associated with those solutions. Advantageous characteristics from a number of existing solutions are combined to form new solutions which in turn are applied to a specific application. A genetic algorithm generally involves creating a systematic encoding nomenclature to describe solutions. Furthermore, a genetic algorithm generally involves providing a mechanism to evaluate solutions. Also, genetic algorithms provide a mechanism to describe characteristics of the various solutions.

In the present system and method, the genetic algorithmic generation is preferably applied to the simulation of VLSI logic circuit blocks. The system and method preferably utilize the test case nomenclature set forth in related application, U.S. Patent Application Serial No. [Attorney Docket Number 10002360-1], entitled “METHOD AND APPARATUS FOR ENCODING AND GENERATING TRANSACTION-BASED STIMULUS FOR SIMULATION OF VLSI CIRCUITS.” Also, the system and method preferably generate a number of original test cases (solutions) as described in the preceding related application. Alternatively, original test cases may be generated randomly or by directed pairing of instructions. This aggregate of test cases is provided to a circuit simulator. The results of the simulator are preferably maintained in a matrix or table. The results preferably detail the

number of times that particular cases cause or stimulate various logic states associated with the simulated VLSI block. The aggregate of solutions and the simulation results are then analyzed by the genetic algorithm. The genetic algorithm preferably identifies states associated with the circuit simulation that have not been produced by the original test cases. The genetic algorithm then combines characteristics of various test cases to generate new test cases. The new test cases are provided to the circuit simulator thereby providing a higher degree of confidence that the entire VLSI chip design has been simulated.

As previously noted, the present system and method preferably utilize the test case generation mechanism utilized in the preceding application. In brief, a test generation algorithm associates transaction types (A, B, C, D, E, F, G, H) with various “strides” (101, 137, 173, 211, 251, 283, 337, 379) as depicted in FIGURE 2. The “strides” are a conceptual construct that define the relative occurrence of various transactions. For example, transaction type A is associated with stride 101. Accordingly, transaction type A is executed via a simulator every 101 simulation cycles. Also, it is important to note that the strides are preferably relatively prime. Specifically, the strides preferably do not possess any common factors. By utilizing strides in this fashion, the relative timing of the various transactions may be iterated in an efficient and comprehensive manner. Moreover, it shall be appreciated that each transaction need not necessarily constitute a single operation on the bus or a single operation state. Instead, each transaction may be associated with a number of states or may be associated with a number of instructions or operations. For example, state A1 may be implemented at simulation cycle 101; state A2 may be implemented at simulation cycle 102; state A3 may be implemented at simulation cycle 103. The number of transaction states may be arbitrarily defined as desired to test a particular VLSI block or other circuit. Also, it shall be appreciated that any number of strides may be utilized. The present description only limits the number of strides to simplify the discussion of the invention.

The circuit simulation process begins by generating test cases. The development of a test case involves assigning particular instructions, operations, instruction states, and/or operation states with particular transaction types of the generation algorithm. For example, transaction type A may be assigned a specific load or write instruction. For descriptive purposes, Test Case 1 is defined by assigning instruction R to stride 101, instruction W to stride 137, and instruction T to stride 173. Accordingly, Test Case 1 may be described using the following format: (TC1:: R:101; W:137; T:173). It shall be appreciated that actual test

cases may comprise any arbitrary number of instructions, operations, instruction states, and/or operation states, which are respectively associated with various strides. The present description merely limits the number of instructions to simplify the attendant discussion.

The instructions associated with various test cases are set forth in FIGURE 4A.

FIGURE 4A illustrates that Test Case 1 has instruction R associated with stride 101, instruction type W associated with stride 137, and instruction type T associated with stride 173. Similarly, Test Case 2 has instruction Q associated with stride 101, instruction R associated with stride 137, and instruction W associated with stride 173. Test Case 3 has instruction M associated with stride 101, instruction N associated with stride 137, and instruction Q associated with stride 173. Test Case N has instruction R associated with stride 101, instruction S associated with stride 137, and instruction T associated with stride 173.

The simulation process begins by injecting the instructions generated by the various test cases into a circuit simulator. The results of the simulation are preferably maintained in a table or matrix. FIGURE 3 illustrates an illustrative table. The rows illustrate all output events that are important to the operation of the logic block. These output events represent logic states associated with the logic block. These output events are monitored to determine whether the logic block functions properly. Any arbitrary number of output events may be monitored.

For exemplary purposes, the following events shall be utilized:

Event 1 = (Ev1) = $J + K + U = (J \text{ and } K \text{ and } U)$;

Event 2 = (Ev2) = $J + K - U = (J \text{ and } K \text{ and not } U)$;

Event 3 = (Ev3) = $K + U = (K \text{ and } U)$; and

Event N = (EvN) = $X + Y + J = (X \text{ and } Y \text{ and } J)$.

The columns of the table are indexed by each test case that is utilized in the circuit simulation. The entries of the table represent the number of times that a particular test case causes a specific event to occur. For illustrative purposes, it has been assumed that an entire set of test cases has been run and a total of all events has been calculated. Accordingly, Test Case 1, as shown in FIGURE 3, did not stimulate the logic state associated with Event 1. However, Test Case 1 stimulated Event 2 ten times. Test Case 1 stimulated Event 3 seven times. Likewise, Test Case 1 stimulated Event N twenty times.

It shall be appreciated that an event may be a subset or a superset of another event. Alternatively, an event may be described as the compliment, negation, or inversion of one or

more logic terms of another similar event. For example, in FIGURE 3, Event 2 = $(J + K - U)$ or descriptively J and K and not U. Also, Event 1 = $(J + K + U)$ or descriptively J and K and U. Clearly, the events are related in that one of the logical variables is inverted. Similarly, Event 3 is $(K + U)$ or descriptively K and U. Consequently, Event 3 is a superset of Event 2.

After tabulating the number of times that the various events have been stimulated by the respective test cases, it is desirable to identify events that have not been stimulated with sufficient frequency to provide an acceptable degree of confidence that the logic block operates appropriately with respect to those events. Accordingly, the identification process may be summarized by the following. First, the result table is searched to identify an event that has not occurred or has occurred infrequently. This event is compared to other events in the table that have occurred with some degree of frequency. Specifically, the logic definition of the non-occurring event is analyzed to determine how closely it matches the logic definition of the more frequently occurring events. Several criteria may be utilized to determine how closely such definitions match. For example, the number of terms that differ between definitions may be examined. Alternatively, the amount of overlap between a superset and a related subset may be examined. This process is iteratively repeated for each event in the table that has not occurred or has occurred infrequently.

In this example, Event 1 has not been stimulated by any of the test cases. However, it is desirable to stimulate that state to fully test the VLSI circuit. In order to construct a test case that has a likelihood of stimulating the non-occurring event (Event 1), the present system and method search the list of events for other logic events that have similar characteristics to the non-occurring event. The underlying assumption is that these similar states are “nearby” in some sense to the non-occurring event. As previously noted, several criteria may be utilized by various software routines to identify acceptable candidate events. In this case, the matching functionality would identify Event 3 as a likely candidate for analysis, since it is a superset of Event 1. Also, the matching functionality would identify Event 2 as a candidate, since its logical definition only differs from the logic definition of Event 1 by a single term. Also, the matching functionality would preferably identify Event N, since its logic definition also requires logic state J.

At this point, it is appropriate to determine which test cases have stimulated the identified similar events (Event 2, Event 3, and Event N). By examining Test Case 1, it is observed that it has stimulated each of the similar events (Event 2, Event 3, and Event N).

Now, Test Case 2 has conversely only stimulated Event N. However, Test Case 2 has caused Event N to occur a significant number of times. Test Case 3 has stimulated both Event 2 and Event 3 two times respectively. Test Case 4 has only stimulated Event 2 five times.

The test cases are examined and their characteristics are weighted. Specifically, the instructions associated with the similar test cases are individually weighted to construct a new test case or test cases. Many weighting algorithms may be implemented by those possessing ordinary skill in the art. Such algorithms may utilize any number of criteria. For example, the closeness or similarity of a particular logic definition of an identified event to the logic definition of the non-occurring event preferably provides a significant weighting factor. In other words, instructions associated with events that are very close to a desired event are weighted more heavily. Also, the relative frequency that a test case with its associated instructions have stimulated the similar events preferably provides a significant weighting factor.

An exemplary weighting algorithm may assign closeness weighting factors of 2, 2, and 1 for Event 2, Event 3, and Event N, respectively, because Event 2 and Event 3 share two logic terms with Event 1 while Event N only shares a single logic term. This closeness weighting factor is multiplied by stimulation occurrence entries for the similar events. The multiplied factor is provided to each instruction by summing across test cases containing the respective instructions. For example, Test Case 1 stimulates Event 2 ten times; Event 3 seven times, and Event N 20 times. Utilizing the respective weighting factors of 2, 2, and 1, this weighting procedure provides a first score of 54. Test Case 2 stimulates Event N fifty times resulting in a second score of 50. Test Case 3 stimulates Event 2 twice, Event 3 twice, and Event N twice, thereby resulting in a third score of 10. Test Case 4 stimulates Event 2 five times, resulting in a fourth score of 10. The first, second, and fourth scores are summed giving a total weight of 114 for instruction R, since it is contained in Test Cases 1, 2, and 4. Similar calculations are performed for instructions W, T, Q, S, M, N to produce respective scores of 104, 64, 58, 10, 10, and 10. For illustrative purposes, an exemplary weighting table is provided in FIGURE 4B.

Now, the weighted instructions may be utilized to construct a new test case or test cases. In actual practice, a large number of instructions will be included in the weighted table, since actual test cases comprise a significant number of instructions in typical implementations. Various methods may be utilized to select constructed test cases from the

weighted table. For example, a random number generation algorithm may be utilized in conjunction with the weighted table. In this situation, each weighted table entry is divided by the sum of all of the weighted table entries. This provides a probability weight for each table entry. The instructions may then be selected randomly according to the probability weights of the instruction entries. It shall be appreciated that a single instruction may be utilized several times in a test case. For example, instruction R received a very high weighted score. Accordingly, it may be appropriate to repeat R by utilizing the test case of (R:101, W:137; R:173). Also, it may be appropriate to assign more heavily weighted instructions to lower strides. It shall further be appreciated that it is appropriate to filter the constructed test cases. Specifically, the constructed test cases should be matched against existing test cases to ensure that a repetition of an already performed simulation does not occur.

Additionally, it is possible to retain timing information related to various instructions. For example, instruction R is associated with stride 101 for Test Case 1 and Test Case 4. Likewise, instruction R is associated with stride 137 for Test Case 2. Since R has been injected into the simulations at a relatively high frequency, it may prove valuable to associate a lower stride with R in the constructed test case. Timing relationships may be selectively associated with instructions utilizing another weighting algorithm. By implementing such an approach, timing information pertaining to the instructions may be retained.

By utilizing the constructed test cases, significant improvement of VLSI simulation may be achieved. Based on an actual simulation with 6000 Events, an initial population of test cases were generated randomly. This random generation succeeded in simulating about 67% of the Events. By applying the genetic algorithm to generate constructed test cases targeted at the remaining non-occurring events, 10% of targeted events were simulated. Accordingly, the use of constructed test cases provides a far greater degree of confidence in the automated circuit simulation process.

When implemented in software, the elements of the present invention are essentially the code segments to perform the necessary tasks. The program or code segments can be stored in a processor readable medium or transmitted by a computer data signal embodied in a carrier wave, or a signal modulated by a carrier, over a transmission medium. The "processor readable medium" may include any medium that can store or transfer information. Examples of the processor readable medium include an electronic circuit, a semiconductor memory device, a ROM, a flash memory, an erasable ROM (EROM), a floppy diskette, a

compact disk CD-ROM, an optical disk, a hard disk, a fiber optic medium, a radio frequency (RF) link, etc. The computer data signal may include any signal that can propagate over a transmission medium such as electronic network channels, optical fibers, air, electromagnetic, RF links, etc. The code segments may be downloaded via computer networks such as the Internet, Intranet, etc.

FIGURE 5 illustrates computer system 500 adapted to use the present invention. Central processing unit (CPU) 501 is coupled to system bus 502. The CPU 501 may be any general purpose CPU, such as an HP PA-8500 or Intel Pentium processor. However, the present invention is not restricted by the architecture of CPU 501 as long as CPU 501 supports the inventive operations as described herein. Bus 502 is coupled to random access memory (RAM) 503, which may be SRAM, DRAM, or SDRAM. ROM 504 is also coupled to bus 502, which may be PROM, EPROM, or EEPROM. RAM 503 and ROM 504 hold user and system data and programs as is well known in the art.

Bus 502 is also coupled to input/output (I/O) controller card 505, communications adapter card 511, user interface card 508, and display card 509. The I/O card 505 connects to storage devices 506, such as one or more of hard drive, CD drive, floppy disk drive, tape drive, to the computer system. Communications card 511 is adapted to couple the computer system 500 to a network 512, which may be one or more of telephone network, local (LAN) and/or wide-area (WAN) network, Ethernet network, and/or Internet network. User interface card 508 couples user input devices, such as keyboard 513 and pointing device 507, to the computer system 500. The display card 509 is driven by CPU 501 to control the display on display device 510.

Although the present invention and its advantages have been described in detail, it should be understood that various changes, substitutions and alterations can be made herein without departing from the spirit and scope of the invention as defined by the appended claims. Moreover, the scope of the present application is not intended to be limited to the particular embodiments of the process, machine, manufacture, composition of matter, means, methods and steps described in the specification. As one of ordinary skill in the art will readily appreciate from the disclosure of the present invention, processes, machines, manufacture, compositions of matter, means, methods, or steps, presently existing or later to be developed that perform substantially the same function or achieve substantially the same result as the corresponding embodiments described herein may be utilized according to the

present invention. Accordingly, the appended claims are intended to include within their scope such processes, machines, manufacture, compositions of matter, means, methods, or steps.

869311.1